**운영체제 텀프로젝트#1 보고서**

2016104156

컴퓨터공학과 이환희

**1. PCB 구현**

<멤버 변수>

int pid : pid를 저장하는 변수

pcb\* pointer : 다음 PCB를 가르키는 포인터 변수

string processState : 프로세스의 상태를 나타내는 스트링 변수

<메소드>

pcb(int pid) : pid로 새로운 PCB를 만들어 내는 생성자

각 변수의 getter,setter

void speakMyState() : 현재 PCB의 PID와 상태를 출력

**2. Queue 구현**

<멤버 변수>

pcb\* head : 큐의 처음을 가르키는 포인터

pcb\* tail : 큐의 마지막을 가르키는 포인터

<메소드>

void push(pcb& p1) : 헤드에 새로 들어오는 p1 가르키게 하고 p1은 기존에 헤드가 가르키던 PCB를 가르키는 식으로 푸쉬

pcb& pop() : tail이 가르키는 PCB를 반환하고 tail을 가르키던 PCB를 tail이 가르킴 (FIFO)

void printAll() : 큐의 모든 노드들을 출력함(PID와 상태)

**3. Main(시나리오) 구현**

Queue jobQueue : 잡큐 선언

Queue readyQueue : 레디큐 선언

Queue keyboardQueue : 키보드큐(디바이스큐) 선언

Queue mouseQueue : 마우스큐(디바이스큐) 선언

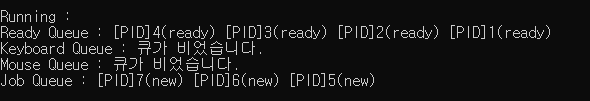
pcb\* runningPCB = NULL : 실행 중인 프로세스(CPU) 가르키는 포인터 변수

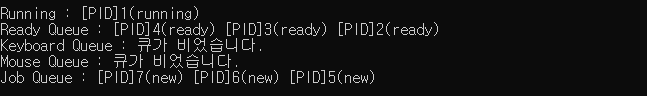
void runningState(Queue& jq, Queue& rq, Queue& dq1, Queue& dq2, pcb\* cpu) : 모든 큐의 상황과 running 중인 프로세스를 출력해줌. (과정을 보기 위해 Sleep()함수 넣어서 출력함.)

**<시나리오>**

**1. 7개의 PCB를 생성한뒤 new상태로 잡큐에 넣어줌.**

**2. 끝에서 부터 순서대로 레디큐에 ready 상태로 넣어줌.**

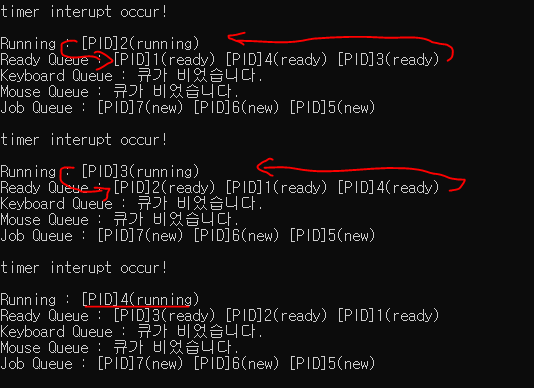


**3. 1번 프로세스 실행**

**4. 타이머 인터럽트 발생(연속 3번)**

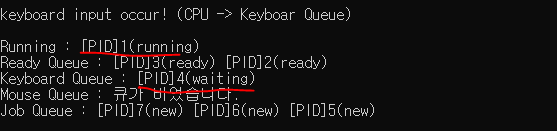
- 1번 다시 레디큐로 가고 2번 실행

- 2번 다시 레디큐로 가고 3번 실행

- 3번 다시 레디큐로 가고 4번 실행

**5. 키보드 입력 대기 발생**

- 실행중이던 4번 waiting 상태로 키보드 큐에 가고 레디큐에 대기하던 1번 실행

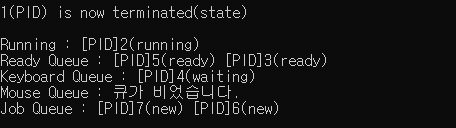


**6. 1번 프로세스 종료**

- 실행중이던 1번 종료메세지 띄우고 delete로 삭제

- 잡큐에서 대기중이던 5번을 ready 상태로 바꾸고 레디큐에 들어감

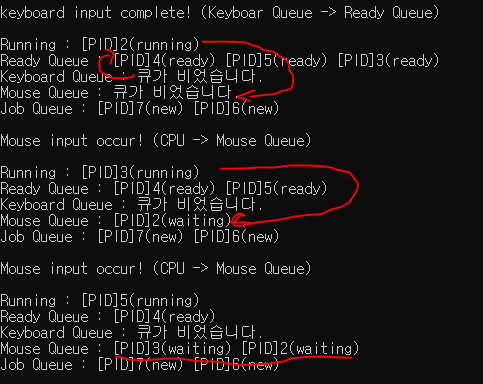
- 레디큐에서 대기하던 2번 실행



**7. 키보드 입력을 완료 받고 마우스 입력 대기(연속 2번)**

- 키보드 입력을 완료해서 키보드 큐에 대기하던 4번이 레디큐로 복귀

- 마우스 입력 대기발생으로 실행중이던 2번이 마우스큐로 들어가고 대기중이던 3번 실행

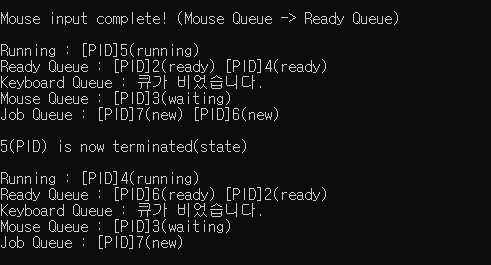
- 마우스 입력 대기 발생으로 실행중이던 3번이 마우스큐로 들어가고 대기중이던 5번 실행

**8. 마우스 입력 완료, 5번 프로세스 종료**

- 마우스 입력 완료된 2번 레디큐로 복귀

- 실행중이던 5번 프로세스 종료

>> 레디큐에 대기중이던 4번 실행

>> 잡큐에 대기중이던 6번 ready 상태로 변경후 레디큐에 들어감

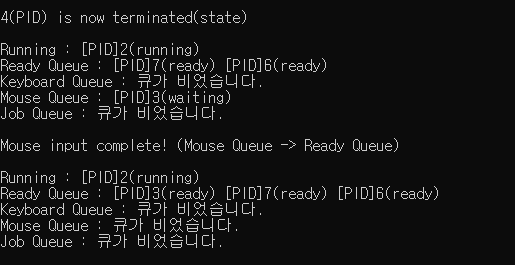
**9. 4번 프로세스 종료, 마우스 입력 완료**

- 실행중이던 4번 프로세스 종료

>> 레디큐에 대기중이던 2번 실행

>> 잡큐에 대기중이던 7번 ready 상태로 변경 후 레디큐에 들어감

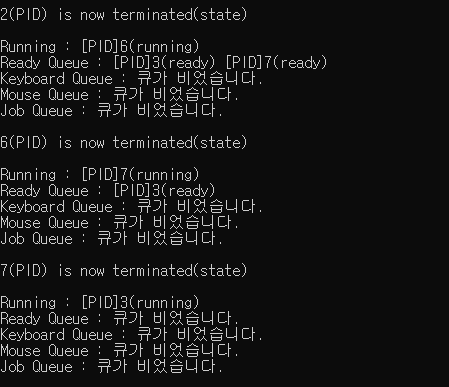
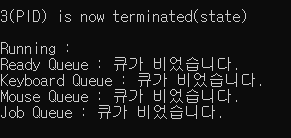
- 마우스 입력 완료로 마우스 큐에 대기중이던 3번 레디큐로 복귀



**10. 모든 프로세스 순차적으로 종료**

- 실행중이던 2번부터 종료

- 레디큐에 대기중인 차례로 실행 >> 종료 반복

- 마지막 3번 종료